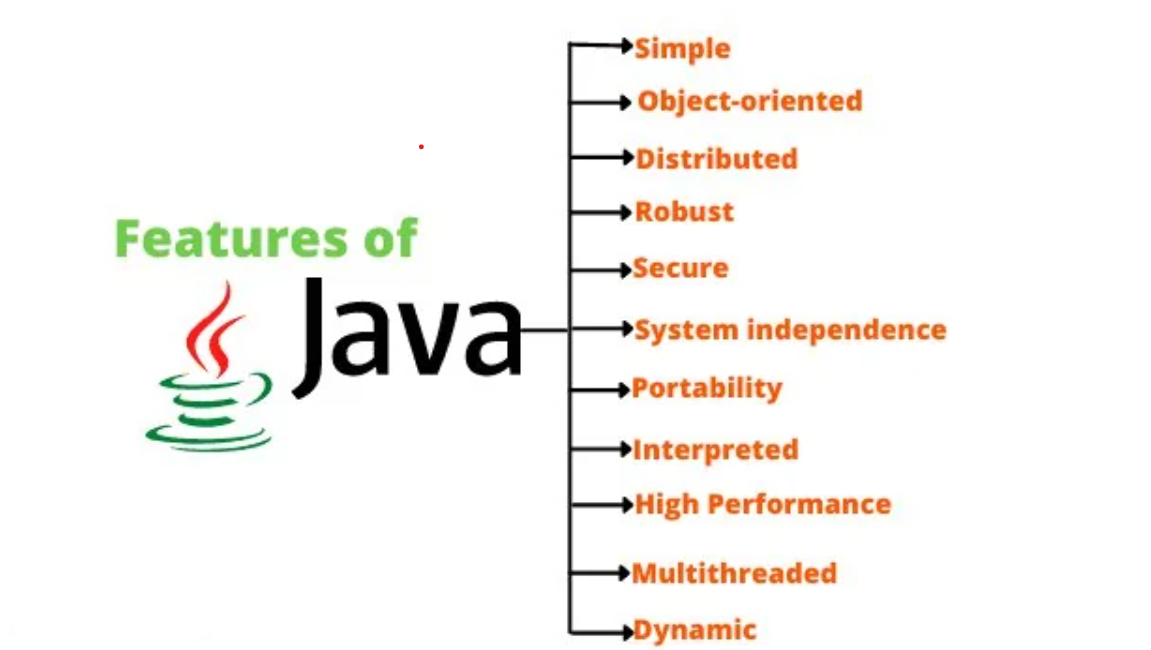
**JAVA Programming:**

* Java is a class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible.
* It is intended to let application developers write once, and run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation.
* Java was first released in 1995 and is widely used for developing applications for desktop, web, and mobile devices.
* Java is known for its simplicity, robustness, and security features, making it a popular choice for enterprise-level applications.
* **JAVA** was developed by James Gosling at **Sun Microsystems** Inc in the year **1995** and later acquired by Oracle Corporation.
* It is a simple programming language. Java makes writing, compiling, and debugging programming easy.
* It helps to create reusable code and modular programs. [Java](https://www.geeksforgeeks.org/java/) is a class-based, object-oriented programming language and is designed to have as few implementation dependencies as possible.
* A general-purpose programming language made for developers to *write once run anywhere* that is compiled Java code can run on all platforms that support Java. Java applications are compiled to byte code that can run on any Java Virtual Machine. The syntax of Java is similar to c/c++.



**History:**

* Java’s history is very interesting. It is a programming language created in 1991.
* James Gosling, Mike Sheridan, and Patrick Naughton, a team of Sun engineers known as the **Green team**initiated the Java language in 1991.
* **Sun Microsystems** released its first public implementation in 1996 as **Java 1.0**. It provides no-cost -run-times on popular platforms.
* Java1.0 compiler was re-written in Java by Arthur Van Hoff to strictly comply with its specifications. With the arrival of Java 2, new versions had multiple configurations built for different types of platforms.
* In 1997, Sun Microsystems approached the ISO standards body and later formalized Java, but it soon withdrew from the process.
* At one time, Sun made most of its Java implementations available without charge, despite their proprietary software status. Sun generated revenue from Java through the selling of licenses for specialized products such as the Java Enterprise System.
* Java platforms are divided into four editions, Java Standard Edition (Java SE), Java Enterprise Edition (Java EE), Java Micro Edition (Java ME), and JavaFX.
* The Java Standard Edition (Java SE) is made up of APIs that provide Java's fundamental features like security, networking, GUI development, etc.
* The Java Enterprise Edition (Java EE) was developed to expand the Java SE with the addition of a collection of specifications that describe frequently used features by commercial applications.
* The Java Micro Edition (Java ME) was developed to facilitate mobile and embedded device applications.
* The JavaFX platform uses a simple API to enable the development of rich online applications.

## Types of Java Overview

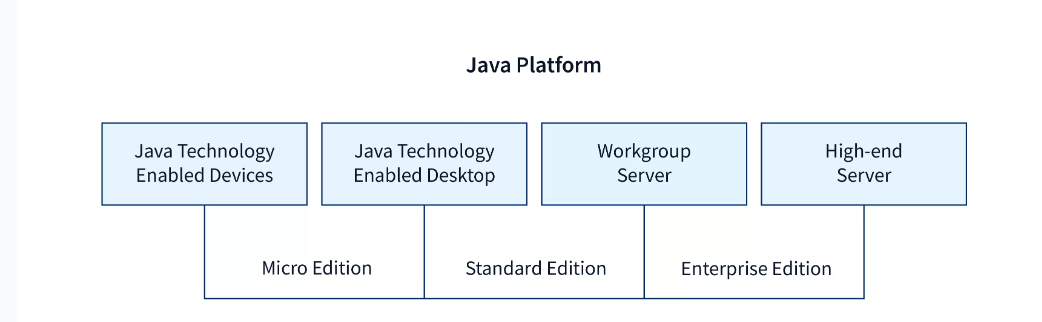
Java technology functions as both a platform and a programming language. High-level object-oriented language Java has a specific syntax and structure. A Java platform is a specific setting where Java programming language programs can be used. Java platforms are divided into four editions, Java Standard Edition (Java SE), Java Enterprise Edition (Java EE), Java Micro Edition (Java ME), and JavaFX.

## What are the Various Types of Java?

**The Four Types of Java:**

1. Java Standard Edition (Java SE),
2. Java Enterprise Edition (Java EE), and
3. Java Micro Edition (Java ME).
4. JavaFX

All these Java editions consist of an API (Application Programming Interface) and JVM (Java Virtual Machine). The tool called the **Java Virtual Machine** (JVM) is used to run Java programs on a certain platform. An API is used to build other software parts or programs. A virtual machine and an API are provided by each Java platform to enable the programs created for that platform to run on any compatible system and take full use of the benefits of the Java programming language.



## The Java Programming Language Platforms

### Java Platform, Standard Edition (Java SE)

The Java Platform Standard Edition (Java SE) is made up of APIs (Application Programming Interfaces) that provide Java's fundamental programming features. It specifies every aspect of the Java programming language, from its fundamental types and objects to its high-level classes for security, Graphical User Interface development, database manipulation, networking, etc.

**The Java SE Platform Includes:**

1. JVM (Java Virtual Machine),
2. JDK (Java Development Kit),
3. JRE (Java Run-time Environment), and
4. Various class libraries.

These components of Java SE are frequently used in Java technology applications in extension to the main APIs.

### Java Platform, Enterprise Edition (Java EE)

The Java Enterprise Edition (Java EE) was developed to expand the Java SE with the addition of a collection of standards/specifications that describe frequently used features by commercial applications. The most recent version of this edition includes more than 40 specifications that assist programmers in building applications that use web services to exchange data, transform object-relational data into entity-relationship models, manage transactional interactions, and other things.

The most used features of Java EE are:

1. **Java Server Pages (JSP):** It is a server-side programming language that makes possible to construct dynamic, device-independent ways to build web-based applications.
2. **Java Server Faces (JSF):** The JSF API offers elements such as inputText, commandButtons, and others and assists in managing their states. Additionally, it offers data translation, server-side validation, etc.
3. **Java Persistence API (JPA):** Java developers get access to an object/relational mapping mechanism through the Java Persistence API, to manage relational data in Java applications.
4. **Enterprise Java Beans (EJB):** It is one of the many Java APIs used for the development of corporate applications. EJB is a server-side software component that defines an application's business logic.

### Java Platform, Micro Edition (Java ME)

Java Micro Edition was developed to facilitate mobile and embedded device applications. This edition is not quite as popular as other editions like Java SE and Java EE, but recently, the Internet of Things (IoT) gadgets have given this technology new hope. The Java ME platform offers an API and a small-footprint virtual machine that helps in running Java ME applications on compact devices.

Some examples of Java ME applications are given below:

1. Calculator for Mobile which demonstrates the functionality of multi-threading.
2. **GPIO** (General Purpose Input Output) applications.
3. Location APIs, etc.

### JavaFX

JavaFX is a platform that provides a modern, hardware-accelerated graphics and media engine for building rich online and desktop applications with a simple and powerful scripting language (JavaFX Script) to enable the development of rich online applications, GUI applications, and Desktop applications. It was created in Java to take the place of Swing as the default GUI library.

Some applications of JavaFX are given below:

1. **NEOS** – New Eurovision Operations System used in the area of Television media.
2. Quote Monitor application is used in the field of finance.
3. **GEONS** Ground System Software (GGSS) Nasa is used in the field of space technology.
4. FORUM Carl Zeiss Meditec AG is used in the field of medicines.

Implementation of a Java application program involves a following step. They include:

1. Creating the program  
2. Compiling the program  
3. Running the program

Remember that, before we begin creating the program, the Java Development Kit (JDK) must be properly installed on our system and also path will be set.

* Creating Program  
     We can create a program using Text Editor (Notepad) or IDE (NetBeans)  
            class Test  
            {  
                       public static void main(String []args)  
                       {  
                                     System.out.println(“My First Java Program.”);  
                       }  
            };

File -> Save -> d:\Test.java

* Compiling the program  
    To compile the program, we must run the Java compiler (javac), with the name of the source file on “command prompt” like as follows

  If everything is OK, the “javac” compiler creates a file called “Test.class” containing byte code of the program.

* Running the program  
     We need to use the Java Interpreter to run a program.

Java programming language is named JAVA. Why?

After the name OAK, the team decided to give it a new name to it and the suggested words were Silk, Jolt, revolutionary, DNA, dynamic, etc. These all names were easy to spell and fun to say, but they all wanted the name to reflect the essence of technology. In accordance with James Gosling, Java the among the top names along with Silk, and since java was a unique name so most of them preferred it.

Java is the name of an island in Indonesia where the first coffee(named java coffee) was produced. And this name was chosen by James Gosling while having coffee near his office. Note that Java is just a name, not an acronym.

**Java Terminology**

Before learning Java, one must be familiar with these common terms of Java.

**1.  Java Virtual Machine(JVM):** This is generally referred to as [JVM](https://www.geeksforgeeks.org/jvm-works-jvm-architecture/#:~:text=JVM(Java%20Virtual%20Machine)%20acts,(Write%20Once%20Run%20Anywhere).). There are three execution phases of a program. They are written, compile and run the program.

* Writing a program is done by a java programmer like you and me.
* The compilation is done by the **JAVAC** compiler which is a primary Java compiler included in the Java development kit (JDK). It takes the Java program as input and generates bytecode as output.
* In the Runningphase of a program,**JVM** executes the bytecode generated by the compiler.

Now, we understood that the function of Java Virtual Machine is to execute the bytecode produced by the compiler. Every Operating System has a different JVM but the output they produce after the execution of bytecode is the same across all the operating systems. This is why Java is known as a**platform-independent language.**

**2. Bytecode in**the **Development Process:**As discussed, the Javac compiler of JDK compiles the java source code into bytecode so that it can be executed by JVM. It is saved as **.class** file by the compiler. To view the bytecode, a disassembler like [javap](https://www.geeksforgeeks.org/javap-tool-in-java-with-examples/) can be used.

**3. Java Development Kit(JDK):**While we were using the term JDK when we learn about bytecode and JVM. So, as the name suggests, it is a complete Java development kit that includes everything including compiler, Java Runtime Environment (JRE), java debuggers, java docs, etc. For the program to execute in java, we need to install JDK on our computer in order to create, compile and run the java program.

**4. Java Runtime Environment (JRE):**JDK includes JRE. JRE installation on our computers allows the java program to run, however, we cannot compile it. JRE includes a browser, JVM, applet support, and plugins. For running the java program, a computer needs JRE.

**5. Garbage Collector:**In Java, programmers can’t delete the objects. To delete or recollect that memory JVM has a program called [Garbage Collector](https://www.geeksforgeeks.org/garbage-collection-java/). Garbage Collectors can recollect the objects that are not referenced. So Java makes the life of a programmer easy by handling memory management. However, programmers should be careful about their code whether they are using objects that have been used for a long time. Because Garbage cannot recover the memory of objects being referenced.

**6. ClassPath:**The [classpath](https://www.geeksforgeeks.org/classpath-in-java/) is the file path where the java runtime and Java compiler look for **.class** files to load. By default, JDK provides many libraries. If you want to include external libraries they should be added to the classpath.

**Primary/Main Features of Java**

**1. Platform Independent:**Compiler converts source code to bytecode and then the JVM executes the bytecode generated by the compiler. This bytecode can run on any platform be it Windows, Linux, or macOS which means if we compile a program on Windows, then we can run it on Linux and vice versa. Each operating system has a different JVM, but the output produced by all the OS is the same after the execution of the bytecode. That is why we call java a platform-independent language.

**2. Object-Oriented Programming Language:**Organizing the program in the terms of a collection of objects is a way of object-oriented programming, each of which represents an instance of the class.

The four main concepts of Object-Oriented programming are:

* Abstraction
* Encapsulation
* Inheritance
* Polymorphism

**3.** **Simple:**Java is one of the simple languages as it does not have complex features like pointers, operator overloading, multiple inheritances, and Explicit memory allocation.

**4.** **Robust:**Java language is robust which means reliable. It is developed in such a way that it puts a lot of effort into checking errors as early as possible, that is why the java compiler is able to detect even those errors that are not easy to detect by another programming language. The main features of java that make it robust are garbage collection, Exception Handling, and memory allocation.

**5.** **Secure:** In java, we don’t have pointers, so we cannot access out-of-bound arrays i.e it shows **ArrayIndexOutOfBound Exception** if we try to do so. That’s why several security flaws like stack corruption or buffer overflow are impossible to exploit in Java. Also, java programs run in an environment that is independent of the os(operating system) environment which makes java programs more secure.

**6.** **Distributed:**We can create distributed applications using the java programming language. Remote Method Invocation and Enterprise Java Beans are used for creating distributed applications in java. The java programs can be easily distributed on one or more systems that are connected to each other through an internet connection.

**7.** **Multithreading:**Java supports multithreading. It is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of the CPU.

**8.** **Portable:**As we know, java code written on one machine can be run on another machine. The platform-independent feature of java in which its platform-independent bytecode can be taken to any platform for execution makes java portable.

**9. High Performance:** Java architecture is defined in such a way that it reduces overhead during the runtime and at some times java uses Just In Time (JIT) compiler where the compiler compiles code on-demand basics where it only compiles those methods that are called making applications to execute faster.

**10. Dynamic flexibility:**Java being completely object-oriented gives us the flexibility to add classes,  new methods to existing classes, and even create new classes through sub-classes. Java even supports functions written in other languages such as C, C++ which are referred to as native methods.

**11. Sandbox Execution:** Java programs run in a separate space that allows user to execute their applications without affecting the underlying system with help of a bytecode verifier. Bytecode verifier also provides additional security as its role is to check the code for any violation of access.

**12. Write Once Run Anywhere:** As discussed above java application generates a ‘.class’ file that corresponds to our applications(program) but contains code in binary format. It provides ease t architecture-neutral ease as bytecode is not dependent on any machine architecture. It is the primary reason java is used in the enterprising IT industry globally worldwide.

**13. Power of compilation and interpretation:** Most languages are designed with the purpose of either they are compiled language or they are interpreted language. But java integrates arising enormous power as Java compiler compiles the source code to bytecode and JVM  executes this bytecode to machine OS-dependent executable code.

class : class keyword is used to declare classes in Java  
public : It is an access specifier. Public means this function is visible to all.  
static : static is again a keyword used to make a function static. To execute a static function you do not have to create an Object of the class. The main() method here is called by JVM, without creating any object for class.  
void : It is the return type, meaning this function will not return anything.  
main : main() method is the most important method in a Java program. This is the method which is executed, hence all the logic must be inside the main() method. If a java class is not having a main() method, it causes compilation error.  
String[] args : This is used to signify that the user may opt to enter parameters to the Java Program at command line. We can use both String[] args or String args[]. Java compiler would accept both forms.

System.out.println : This is used to print anything on the console like “printf” in C language.

**Example:**

**// Importing classes from packages**

**import java.io.\*;**

**// Main class**

**public class GFG {**

**// Main driver method**

**public static void main(String[] args)**

**{**

**// Print statement**

**System.out.println("Welcome to JAVA");**

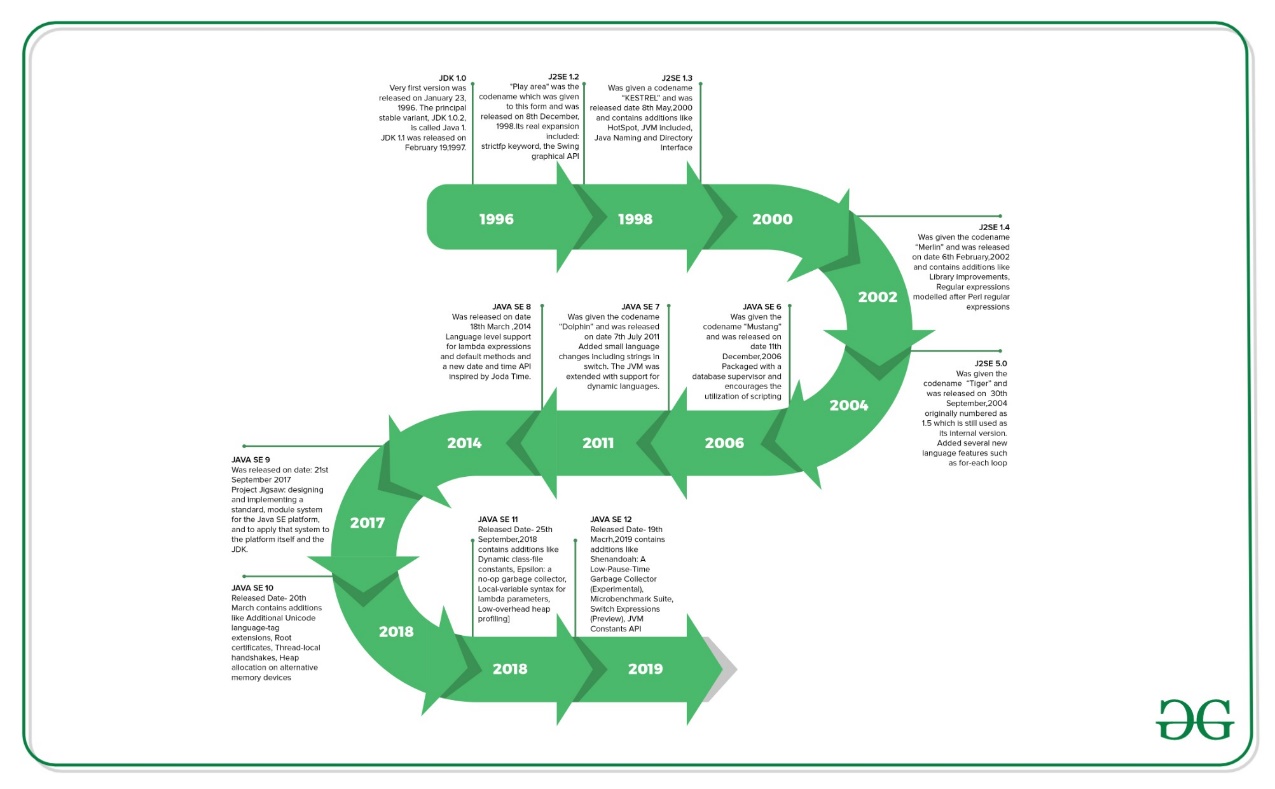
**}**

**}**

[Java](https://www.geeksforgeeks.org/java-tutorials/) is an [Object-Oriented programming](https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/) language developed by **James Gosling** in the early 1990s. The team initiated this project to develop a language for digital devices such as set-top boxes, television, etc. Originally [C++](https://www.geeksforgeeks.org/c-plus-plus/) was considered to be used in the project but the idea was rejected for several reasons(For instance C++ required more memory). Gosling endeavoured to alter and expand [C++](https://www.geeksforgeeks.org/c-plus-plus/) however before long surrendered that for making another stage called **Green**. James Gosling and his team called their project “**Greentalk**” and its file extension was **.gt** and later became to known as “**OAK**”.

**Why “Oak”?**

The name **Oak** was used by **Gosling** after an **oak tree** that remained outside his office. Also, Oak is an image of solidarity and picked as a national tree of numerous nations like the U.S.A., France, Germany, Romania, etc. But they had to later rename it as “**JAVA**” as it was already a trademark by **Oak Technologies**. **“JAVA”** Gosling and his team did a brainstorm session and after the session, they came up with several names such as **JAVA, DNA, SILK, RUBY, etc.** **Java** name was decided after much discussion since it was so unique. The name Java originates from a sort of **espresso bean**, Java. Gosling came up with this name while having a coffee near his office. Java was created on the principles like **Robust, Portable, Platform Independent, High Performance, Multithread, etc.** and was called one of the **Ten Best Products of 1995** by the **TIME MAGAZINE**. Currently, Java is used in **internet programming, mobile devices, games, e-business solutions, etc.**

The [Java language](https://www.geeksforgeeks.org/java/) has experienced a few changes since **JDK 1.0** just as various augmentations of classes and packages to the standard library. In Addition to the language changes, considerably more sensational changes have been made to the Java Class Library throughout the years, which has developed from a couple of hundred classes in JDK 1.0 to more than three thousand in J2SE 5.**History of various Java versions:**

| **Version** | **Release Date** | **Major changes** |
| --- | --- | --- |
| JDK Beta | 1995 |  |
| JDK 1.0 | January 1996 | The Very first version was released on January 23, 1996. The principal stable variant, JDK 1.0.2, is called Java 1. |
| JDK 1.1 | February 1997 | Was released on February 19, 1997. There were many additions in JDK 1.1 as compared to version 1.0 such as   * A broad retooling of the AWT occasion show * Inner classes added to the language * JavaBeans * JDBC * RMI |
| J2SE 1.2 | December 1998 | “Play area” was the codename which was given to this form and was released on 8th December 1998. Its real expansion included: strictfp keyword   * the Swing graphical API was coordinated into the centre classes * Sun’s JVM was outfitted with a JIT compiler out of the blue * Java module * Java IDL, an IDL usage for CORBA interoperability * Collections system |
| J2SE 1.3 | May 2000 | Codename- “KESTREL” Release Date- 8th May 2000 Additions:   * HotSpot JVM included * Java Naming and Directory Interface * JPDA * JavaSound * Synthetic proxy classes |
| J2SE 1.4 | February 2002 | Codename- “Merlin” Release Date- 6th February 2002 Additions: Library improvements   * Regular expressions modelled after Perl regular expressions * The image I/O API for reading and writing images in formats like JPEG and PNG * Integrated XML parser and XSLT processor (JAXP) (specified in JSR 5 and JSR 63) * Preferences API (java.util.prefs)   Public Support and security updates for this version ended in October 2008. |
| J2SE 5.0 | September 2004 | Codename- “Tiger” Release Date- “30th September 2004” Originally numbered as 1.5 which is still used as its internal version. Added several new language features such as:   * for-each loop * Generics * Autoboxing * Var-args |
| JAVA SE 6 | December 2006 | Codename- “Mustang” Released Date- 11th December 2006 Packaged with a database supervisor and encourages the utilization of scripting languages with the JVM. Replaced the name J2SE with java SE and dropped the .0 from the version number. Additions:   * Upgrade of JAXB to version 2.0: Including integration of a StAX parser. * Support for pluggable annotations (JSR 269). * JDBC 4.0 support (JSR 221) |
| JAVA SE 7 | July 2011 | Codename- “Dolphin” Release Date- 7th July 2011 Added small language changes including strings in the switch. The JVM was extended with support for dynamic languages. Additions:   * Compressed 64-bit pointers. * Binary Integer Literals. * Upstream updates to XML and Unicode. |
| JAVA SE 8 | March 2014 | Released Date- 18th March 2014 Language level support for lambda expressions and default methods and a new date and time API inspired by Joda Time. |
| JAVA SE 9 | September 2017 | Release Date: 21st September 2017 Project Jigsaw: designing and implementing a standard, a module system for the Java SE platform, and to apply that system to the platform itself and the JDK. |
| JAVA SE 10 | March 2018 | Released Date- 20th March Addition:   * Additional Unicode language-tag extensions * Root certificates * Thread-local handshakes * Heap allocation on alternative memory devices * Remove the native-header generation tool – javah. * Consolidate the JDK forest into a single repository. |
| JAVA SE 11 | September 2018 | Released Date- 25th September, 2018 Additions-   * Dynamic class-file constants * Epsilon: a no-op garbage collector * The local-variable syntax for lambda parameters * Low-overhead heap profiling * HTTP client (standard) * Transport Layer Security (TLS) 1.3 * Flight recorder |
| JAVA SE 12 | March 2019 | Released Date- 19th March 2019 Additions-   * Shenandoah: A Low-Pause-Time Garbage Collector (Experimental) * Microbenchmark Suite * Switch Expressions (Preview) * JVM Constants API * One AArch64 Port, Not Two * Default CDS Archives |
| JAVA SE 13 | September 2019 | Released Date – 17th September 2019 Additions-   * Text Blocks (Multiline strings). * Switch Expressions. * Enhanced Thread-local handshakes. |
| JAVA SE 14 | March 2020 | Released Date – 17th March 2020 Additions-   * Records (new class type for data modeling). * Pattern Matching for instanceof. * Helpful NullPointerExceptions. |
| JAVA SE 15 | September 2020 | Released Date – 15th September 2020 Additions-   * Sealed Classes. * Hidden Classes. * Foreign Function and Memory API (Incubator). |
| JAVA SE 16 | March 2021 | Released Date – 16th March 2021 Additions-   * Records (preview feature). * Pattern Matching for switch (preview feature). * Unix Domain Socket Channel (Incubator). |
| JAVA SE 17 | September 2021 | Released Date – 14th September 2021 Additions-   * Sealed Classes (finalized). * Pattern Matching for instanceof (finalized). * Strong encapsulation of JDK internals by default. * New macOS rendering pipeline. |

These three programming languages are the most popular among coders in terms of competitive coding and programming. C++ of today in its efficiency, speed, and memory makes it widely popular among coders. Java is platform-independent. It continues to add considerable value to the world of software development. Python requires less typing and provides new libraries, fast prototyping, and several other new features. Let’s look at the comparison between these popular coding languages.

**C++ Vs Java:**

| **TOPIC** | **C++** | **Java** |
| --- | --- | --- |
| Memory Management | Use of pointers, structures, union | No use of pointers. Supports references, thread and interfaces. |
| Libraries | Comparatively available with low-level functionalities | Wide range of classes for various high-level services |
| Multiple Inheritance | Provide both single and multiple inheritance. | Multiple inheritances is partially done through interfaces |
| Operator Overloading | Supports operator overloading | It doesn’t support this feature |
| Program Handling | Functions and variables can reside outside classes. | Functions and variables reside only in classes, packages are used. |
| Portability | Platform dependent, must be recompiled for different platform | Platform independent, byte code generated works on every OS. |
| Thread Support | No built-in support for threads, depends on libraries. | It has built-in thread support. |